SPPU PRACTICAL EXAMINATION - PROBLEM STATEMENTS

Problem Statement 1 (CRUD)

Create collection Student with fields as Roll_No, Name, Class, Marks, Address, Enrolled_Courses.
(Hint: One student can enrol in multiple courses. Use Array to store the names of courses enrolled)
Insert 10 documents in the collection Student. Write the queries for following.
1. List the names of students who have enrolled in the course “DBMS”, “TOC”.
2. List the Roll numbers and class of students who have marks more than 50 or class as TE.
3. Update the entire record of roll_no A10.
4. Display the names of students having 3rd and 4th highest marks.
5. Delete the records of students having marks less than 20.
6. Delete only first record from the collection.

Problem Statement 2 (DDL USING MYSQL)

Create following tables using a given schema and insert appropriate data into the same:

Customer (CustID, Name, Cust_Address, Phone_no, Email_ID, Age)
Branch (Branch ID, Branch_Name, Address)
Account (Account_no, Branch ID, CustID, open_date, Account_type, Balance)

1. Create the tables with referential integrity.
2. Draw the ER diagram for the same.
3. Create a View as Saving account displaying the customer details with the open date as 16/8/2018.
4. Update the View with Cust_Address as Pune for CustID =103.
5. Create a View as Loan account displaying the customer details with the open date as 16/2/2018.
6. Create an Index on primary key column of table Customer.
7. Create an Index on primary key column of table Branch.
8. Create a sequence on Customer Table.
9. Create synonym ‘Cust_info’ for branch table.

Problem Statement 3 (Unnamed Block)

Employee(emp_id, dept_idemp_name, DoJ, salary, commission,job_title)

Salary_Increment(emp_id, new_salary)

Consider the schema given above. Write a PLSQL Unnamed Block of code to increase the salary of employee 115 based on the following conditions:

Accept emp_id from user. If experience of employee is more than 10 years, increase salary by 20%. If experience is greater than 5 years, increase salary by 10% Otherwise 5%. (Hint: Find the years of experience from Date of Joining (DoJ)). Store the incremented salary in Salary_Increment table.

Also handle the exception by named exception handler or user defined exception handler.

Problem Statement 4 (AGGREGATION & INDEXING USING MONGODB)

Create the Collection Student_Data(Student _ID, Student _Name, Department, Marks)and solve the following:

1. Display all Students based on their departments along with an average Marks of a particular department.
2. Displays the number of Students associated along with a particular department.
3. Display list of Students with the highest Marks in each Department in descending order of Marks.
4. Create an index on field Student_ID.
5. Create an index on fields "Student_Name' and "Department".
6. Drop an index on field Student_ID.
7. Drop an index on fields "Student_Name' and "Department".

Problem Statement 5 (DML USING MYSQL)

Create following tables using a given schema and insert appropriate data into the same:

Customer (CustID, Name, Cust_Address, Phone_no, Age)
Branch (Branch ID, Branch_Name, Address)
Account (Account_no, Branch ID, CustID, date_open, Account_type, Balance)

1. Add the column "Email_Address" in Customer table.
2. Change the name of column "Email_Address" to "Email_ID" in Customer table.
3. Display the customer details with highest balance in the account.
4. Display the customer details with lowest balance for account type= "Saving Account".
5. Display the customer details that live in Pune and have age greater than 35.
6. Display the Cust_ID, Name and Age of the customer in ascending order of their age.
7. Display the Name and Branch ID of the customer group by the Account_type.

Problem Statement 6 (Procedures / Functions)

Employee(emp_id, dept_idemp_name, DoJ, salary, commission,job_title)

1. Consider the schema given above. Keep the commission column empty initially. Write a **Stored Procedure** to record the employee commission based on following conditions. If salary is more than 10000 then commission is 0.4%, if Salary is less than 10000 but experience is more than 10 years then 0.35%, if salary is less than 3000 then commission is 0.25%. In the remaining cases commission is 0.15%.

2. Write a PLSQL **Function** that takes department ID and returns the name of the manager of the department.

Problem Statement 7 (Map Reduce)

Create Book Collection with (Title, Author_name, Borrowed_status) as fields. Write Map Reduce Functions for following requirements.

1. Display Author wise list of books.
2. Display Author wise list of books having Borrowed status as "True".
3. Display Author wise list of books having price greater than 300.

Problem Statement 8 (JOINS & SUBQUERIES USING MYSQL)

Consider Following Schema

Employee (Employee_id, First_name, last_name , hire_date, salary, Job_title, manager_id, department_id)

Departments(Department_id, Department_name, Manager_id, Location_id)

Locations(location_id ,street_address ,postal_code, city, state, country_id)

Manager(Manager_id, Manager_name)

Create the tables with referential integrity. Solve following queries using joins and subqueries.

1. Write a query to find the names (first_name, last_name) and the salaries of the employees who have a higher salary than the employee whose last_name="Singh”.
2. Write a query to find the names (first_name, last_name) of the employees who have a manager and work for a department based in the United States.
2. Write a query to find the names (first_name, last_name), the salary of the employees whose salary is greater than the average salary.
3. Write a query to find the employee id, name (last_name) along with their manager_id, manager name (last_name).
4. Find the names and hire date of the employees who were hired after 'Jones'.

Problem Statement 9 (Cursors)

Consider a table Employee with schema as Employee (Emp_id, Emp_Name,Salary).

1. Write an explicit cursor to display records of all employees with salary greater than 50,000.

2. Write a PL/SQL block of code using Implicit Cursor that will display total number of tuples in Employee table.
3. Write a PL/SQL block of code using Parameterized Cursor that will display salary of employee id entered by the user.

Problem Statement 10 (CRUD)

Create a collection Social_Media having fields as User_Id, User_Name, No_of_Posts, No_of_Friends, Friends_List, Interests. (Hint: Friends_List and Interests can be of array type)
Insert 20 documents in the collection Social_Media. Write queries for following.
1. List all the users from collection Social_Media in formatted manner.
2. Find all users having number of posts greater than 100.
3. List the user names and their respective Friens_List
4. Display the user ids and Friends list of users who have more than 5 friends.
5. Display all users with no of posts in descending order.

Problem Statement 11 (DDL USING MYSQL)

Create following tables using a given schema and insert appropriate data into the same:

Customer (CustID, Name, Cust_Address, Phone_no, Email_ID, Age)
Branch (Branch ID, Branch_Name, Address)
Account (Account_no, Branch ID, CustID, date_open, Account_type, Balance)

1. Create the tables with referential integrity.
2. Draw the ER diagram for the same.
3. Create an Index on primary key column of table Account
4. Create the view as Customer_Info displaying the customer details for age less than 45.
5. Update the View with open date as 16/4/2017
6. Create a sequence on Branch able.
7. Create synonym 'Branch_info' for branch table.

Problem Statement 12 (Triggers)

Employee(emp_id, dept_idemp_name, DoJ, salary, commission,job_title)

Consider the schema given above for Write a PLSQL Program to

1. Create a Trigger to ensure the salary of the employee is not decreased.

2. Whenever the job title is changed for an employee write the following details into job_history table. Employee ID, old job title, old department ID, DoJ for start date, system date for end date.

Problem Statement 13 (Map Reduce)

Create collection for Student{roll_no, name, class, dept, aggregate_marks}. Write Map Reduce Functions for following requirements.

1. Finding the total marks of students of "TE" class department-wise.
2. Finding the highest marks of students of "SE" class department-wise.
3. Find Average marks of students of "BE" class department-wise.

Problem Statement 14 (DML USING MYSQL)

Create following tables using a given schema and insert appropriate data into the same:

Customer (CustID, Name, Cust_Address, Phone_no, Email_ID, Age)
Branch (Branch ID, Branch_Name, Address)
Account (Account_no, Branch ID, CustID, date_open, Account_type, Balance)

1. Modify the size of column "Email_Address" to 20 in Customer table.
2. Change the column "Email_Address" to Not Null in Customer table.
3. Display the total customers with the balance >50, 000 Rs.
4. Display average balance for account type="Saving Account".
5. Display the customer details that lives in Pune or name starts with 'A'.
6. Create a table Saving_Account with (Account_no, Branch ID, CustID, date_open, Balance) using Account Table.
7. Display the customer details Age wise with balance>=20,000Rs

Problem Statement 15 (Cursors)

Consider the following schema for Products table.

Products(Product_id, Product_Name, Product_Type, Price)

1. Write a parameterized cursor to display all products in the given price range of price and type 'Apparel'.

Hint: Take the user input for minimum and maximum price for price range.

2. Write an explicit cursor to display information of all products with Price greater than 5000.

3. Write an implicit cursor to display the number of records affected by the update operation incrementing Price of all products by 1000.

Problem Statement 16

Implement MongoDB database connectivity with Java.

Problem Statement 17 (JOINS & SUBQUERIES USING MYSQL)

Consider Following Schema

Employee (Employee_id, First_name, Last_name , Hire_date, Salary, Job_title, Manager_id, department_id)

Departments(Department_id, Department_name, Manager_id, Location_id)

Locations(Location_id , Street_address , Postal_code, city, state, Country_id)

Manager(Manager_id, Manager_name)

Create the tables with referential integrity.Solve following queries using joins and subqueries.

1. Write a query to find the names (first_name, last_name), the salary of the employees who earn more than the average salary and who works in any of the IT departments.
2. Write a query to find the names (first_name, last_name), the salary of the employees who earn the same salary as the minimum salary for all departments.
3. Write a query to display the employee ID, first name, last names, salary of all employees whose salary is above average for their departments.
4. Write a query to display the department name, manager name, and city.
5. Write a query to display the name (first_name, last_name), hire date, salary of all managers whose experience is more than 15 years.

Problem Statement 18 (Procedures / Functions)

Consider following schema for Bank database.

Account(Account_No, Cust_Name, Balance, NoOfYears)

Earned_Interest(Account_No, Interest_Amt)

1. Write a PL/SQL **procedure** for following requirement. Take as input Account_No and Interest Rate from User. Calculate the Interest_Amt as simple interest for the given Account_No and store it in Earned_Interest table. Display all the details of Earned_Interest Table.

2. Write a PLSQL function to display all records from Account table having Balance greater than 50,000.

Problem Statement 19 (Aggregation & Indexing)

Create the Collection Movies_Data(Movie_ID, Movie_Name, Director, Genre, BoxOfficeCollection) **and solve the following:**

1. Display a list stating how many Movies are directed by each “Director”.
2. Display list of Movies with the highest BoxOfficeCollection in each Genre.
3. Display list of Movies with the highest BoxOfficeCollection in each Genre in ascending order of BoxOfficeCollection.
4. Create an index on field Movie_ID.
5. Create an index on fields ” Movie_Name” and ” Director”.
6. Drop an index on field Movie_ID.
7. Drop an index on fields ” Movie_Name” and ” Director”.

Problem Statement 20

Implement MYSQL database connectivity with PHP.

Problem Statement 21 (Triggers)

Employee(emp_id, emp_name, salary, designation)

Salary_Backup(emp_id, old_salary, new_salary, salary_difference)

Create a Trigger to record salary change of the employee. Whenever salary is updated insert the details in Salary_Backup table.

Create a Trigger that will prevent deleting the employee record having designation as CEO.
